W3C

SOAP Message Transmission Optimization Mechanism

W3C Recommendation 25 January 2005

This version:
> http://www.w3.org/TR/2005/REC-soap12-mtom-20050125/

Latest version:
> http://www.w3.org/TR/soap12-mtom/

Previous version:
> http://www.w3.org/TR/2004/PR-soap12-mtom-20041116/

Editors:
> Martin Gudgin, Microsoft
> Noah Mendelsohn, IBM
> Mark Nottingham, BEA
> Hervé Ruellan, Canon

Please refer to the **errata** for this document, which may include normative corrections.

See also **translations**.

Abstract

This document describes an abstract feature and a concrete implementation of it for optimizing the transmission and/or wire format of SOAP messages. The concrete implementation relies on the [XML-binary Optimized Packaging] format for carrying SOAP messages.

Status of this Document

This section describes the status of this document at the time of its publication. Other documents may supersede this document. A list of current W3C publications

and the latest revision of this technical report can be found in the W3C technical reports index at http://www.w3.org/TR/.

This document is a Recommendation of the W3C. It has been reviewed by W3C Members and other interested parties and has been endorsed by the Director as a W3C Recommendation. It is a stable document and may be used as reference material or cited as a normative reference from another document. W3C's role in making the Recommendation is to draw attention to the specification and to promote its widespread deployment. This enhances the functionality and interoperability of the Web.

This document has been produced by the XML Protocol Working Group (WG) as part of the W3C Web Services Activity. The English version of this specification is the only normative version. However, for translations of this document, see http://www.w3.org/2003/03/Translations/byTechnology?technology=soap12-mtom.

Please report errors in this document to xmlp-comments@w3.org (archive). The errata list for this edition is available at http://www.w3.org/2005/01/soap12-mtom-errata

This document is based upon the SOAP Message Transmission Optimization Mechanism Proposed Recommendation of 16 November 2004. Feedback received during that review resulted in no changes. Evidence of interoperation between at least two implementations of this specification are documented in the Implementation Summary. Changes between these two versions are described in a diff document.

This document has been produced under the 24 January 2002 CPP as amended by the W3C Patent Policy Transition Procedure. An individual who has actual knowledge of a patent which the individual believes contains Essential Claim(s) with respect to this specification should disclose the information in accordance with section 6 of the W3C Patent Policy. Patent disclosures relevant to this specification may be found on the Working Group's patent disclosure page.

A list of current W3C Recommendations and other technical documents can be found at http://www.w3.org/TR/.

Table of Contents

Appendices

---

1 Introduction

The first part of this document (**2 Abstract SOAP Transmission Optimization Feature**) describes an abstract feature for optimizing the transmission and/or wire format of a SOAP message ([SOAP Version 1.2 Part 1: Messaging Framework]) by selectively encoding portions of the message, while still presenting an XML Infoset to the SOAP application.

Use of the Abstract SOAP Transmission Optimization Feature is a hop-by-hop contract between a SOAP node and the next SOAP node in the SOAP message path, providing no mandatory convention for optimization of SOAP transmission through intermediaries. The feature does provide optional means by which binding implementations MAY choose to facilitate the efficient pass-through of optimized data contained within headers or bodies relayed by an intermediary (see **2.3.4 Binding Optimizations at Intermediaries**). Additional specifications might also be written to provide for other optimized multi-hop capabilities, perhaps building on the mechanisms provided herein.

The second part (**3 An Optimized MIME Multipart/Related Serialization of SOAP Messages**) describes an Optimized MIME Multipart/Related Serialization of SOAP Messages implementing the Abstract SOAP Transmission Optimization Feature in a binding independent way. This implementation relies on the [XML-binary Optimized Packaging] format.

The third part (**4 HTTP SOAP Transmission Optimization Feature**) uses this Optimized MIME Multipart/Related Serialization of SOAP Messages for describing an implementation of the Abstract Transmission Optimization Feature for the SOAP 1.2 HTTP binding (see [SOAP Version 1.2 Part 2: Adjuncts] 7. SOAP HTTP Binding).

1.1 Notational Conventions

The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC 2119].

This specification uses a number of namespace prefixes throughout; they are listed in **[Prefixes and Namespaces used in this specification.]**. Note that the choice of any namespace prefix is arbitrary and not semantically significant (see XML Infoset [XMLInfoSet]).

Prefixes and Namespaces used in this specification.

Prefix	Namespace
	Notes
env	"http://www.w3.org/2003/05/soap-envelope"
	A normative XML Schema [XML Schema Part 1: Structures Second Edition], [XML Schema Part 2: Datatypes Second Edition] document for the "http://www.w3.org/2003/05/soap-envelope" namespace can be found at http://www.w3.org/2003/05/soap-envelope.
xop	"http://www.w3.org/2004/08/xop/include"
	A non-normative XML Schema [XML Schema Part 1: Structures Second Edition], [XML Schema Part 2: Datatypes Second Edition] document for the "http://www.w3.org/2004/08/xop/include" namespace can be found at http://www.w3.org/2004/08/xop/include.
rep	"http://www.w3.org/2004/08/representation"
	A normative XML Schema [XML Schema Part 1: Structures Second Edition], [XML Schema Part 2: Datatypes Second Edition] document for the "http://www.w3.org/2004/08/representation" namespace can be found at TBD.
xs	"http://www.w3.org/2001/XMLSchema"
	The namespace of XML Schema data types (see [XML Schema Part 2: Datatypes Second Edition]).

1.2 Relation to other specifications

The **4 HTTP SOAP Transmission Optimization Feature** (which is an

implementation of the **2 Abstract SOAP Transmission Optimization Feature** for the SOAP 1.2 HTTP binding) is intended to enhance the SOAP HTTP binding described in [SOAP Version 1.2 Part 2: Adjuncts] 7. SOAP HTTP Binding or an updated version of it.

This document along with [XML-binary Optimized Packaging] and [SOAP Representation Header] has been produced in conjunction with the development of requirements embodied in the [W3C.soap-attachment-req] document.

2 Abstract SOAP Transmission Optimization Feature

2.1 Introduction

The Abstract SOAP Transmission Optimization Feature enables SOAP bindings to optimize the transmission and/or wire format of a SOAP message by selectively encoding portions of the message, whilst still presenting an XML Infoset to the SOAP application. Optimization is available only for element content that is in a canonical lexical representation of the `xs:base64Binary` data type (see [XML Schema Part 2: Datatypes Second Edition] 3.2.16 base64Binary).

Note: because there is a one-to-one correspondence between such canonical forms and values in the value space of `xs:base64Binary`, MTOM implementations typically optimize by transmitting a compact representation of the value in place of the less compact character sequence. At the receiver, the character form can be reconstructed if necessary.

Consistent with the requirement that a SOAP binding transmit the Envelope Infoset intact, implementations of this feature MUST NOT substitute canonical for non-canonical representations, or make other changes to optimized element content.

2.2 Abstract SOAP Transmission Optimization Feature Name

This Abstract SOAP Transmission Optimization Feature is identified by the URI:

- "http://www.w3.org/2004/08/soap/features/abstract-optimization".

2.3 Abstract SOAP Transmission Optimization Feature Processing

2.3.1 Sending a message

When sending a SOAP Message, if the Abstract Transmission Optimization Feature is used in combination with the SOAP Request-Response Message Exchange Pattern ([SOAP Version 1.2 Part 2: Adjuncts] 6.2 SOAP Request-Response Message Exchange Pattern) or the SOAP Response Message

Exchange Pattern ([SOAP Version 1.2 Part 2: Adjuncts] 6.3 SOAP Response
Message Exchange Pattern), the
`http://www.w3.org/2003/05/soap/mep/OutboundMessage` property is the Infoset of
the SOAP Message to be sent. Similar rules should be applied for other MEPs, as
appropriate.

The purpose of the Abstract SOAP Transmission Optimization Feature is to
optimize the transmission of base64 encoded data. To be optimized, the
characters comprising the [children] of an *element information item* MUST be in
the canonical form of `xs:base64Binary` (see [XML Schema Part 2: Datatypes
Second Edition] 3.2.16 base64Binary) and MUST NOT contain any whitespace
characters, preceding, inline with or following the non-whitespace content.

Note: the means of identifying *element information items* that contain base64
encoded data in canonical lexical form are implementation-dependent. Some
implementations can identify such *element information items* by construction (e.g.,
because a certain API may create only canonical forms); others may check the
characters prior to sending, others may rely on information in the description such
as the presence and/or value of the `xmlmime:expectedMediaType` schema
annotation (see [Assigning Media Types to Binary Data in XML]), if a schema is
available. Because of the need to exactly preserve the characters in the
transmitted Infoset, non-canonical representations MUST NOT be optimized by
implementations of this feature.

2.3.2 Receiving a message

When receiving a SOAP message optimized using an implementation of the
Abstract SOAP Transmission Optimization Feature, a SOAP node SHOULD
generate a fault if it does not support the implementation used or the Abstract
SOAP Transmission Optimization Feature.

Upon reception of an optimized SOAP message, the receiving node MUST
reconstruct an Envelope Infoset from the optimized SOAP message. Then, the
receiving node MUST perform SOAP processing on the reconstructed Infoset (see
[SOAP Version 1.2 Part 1: Messaging Framework] 2. SOAP Processing Model). In
all cases, the received Infoset MUST be exactly the same as that transmitted by
the sender.

Implementations are free to reconstruct only those portions actually needed for
processing, or to present information from the message in a form convenient for
efficient processing. For example, a value sent in an optimized form (e.g., binary)
MAY be made available in that form as well as in the base64 encoded character
form.

When this feature is used in combination with the SOAP Request-Response
Message Exchange Pattern ([SOAP Version 1.2 Part 2: Adjuncts] 6.2 SOAP

Request-Response Message Exchange Pattern) or the SOAP Response Message
Exchange Pattern ([SOAP Version 1.2 Part 2: Adjuncts] 6.3 SOAP Response
Message Exchange Pattern), the Infoset contained in the
`http://www.w3.org/2003/05/soap/mep/InboundMessage` property is the Infoset of
the reconstructed SOAP Envelope. Similar rules should be applied for other
MEPs, as appropriate.

2.3.3 Intermediaries

Use of the Abstract SOAP Transmission Optimization Feature is a hop-by-hop
contract between a SOAP node and the next SOAP node in the SOAP message
path. Therefore, no changes or restrictions to the SOAP processing model are
introduced by this feature at an intermediary. The section **2.3.4 Binding
Optimizations at Intermediaries** details the means by which certain optimizations
can be performed by bindings at intermediaries.

However a SOAP intermediary implementing the Abstract Transmission
Optimization Feature MUST still follow the rules related to the usage of an
implementation of the Abstract Transmission Optimization Feature when receiving
the message (see **2.3.2 Receiving a message**) and those related to the usage of
an implementation of the Abstract SOAP Transmission Optimization Feature when
sending the message (see **2.3.1 Sending a message**). In particular, it MUST
follow the rules for relaying SOAP messages (see [SOAP Version 1.2 Part 1:
Messaging Framework] 2.7 Relaying SOAP Messages).

2.3.4 Binding Optimizations at Intermediaries

As described in [SOAP Version 1.2 Part 1: Messaging Framework] 2.7 Relaying
SOAP Messages, a SOAP intermediary may be called upon to relay intact certain
headers, or to reinsert headers identical to those received and removed for
processing. Furthermore, many intermediaries will relay unmodified the contents of
the SOAP body. In all these cases, portions of the relayed message have content
identical to corresponding portions of the inbound message.

The Abstract SOAP Transmission Optimization Feature does not require any
particular correspondence between the optimization of the inbound message and
the outbound message, even when optimized portions of the inbound message
are relayed intact, or reinserted in identical form in the envelope Infoset.
Nonetheless, the implementations of the receiving binding and the binding used to
transmit the relayed message MAY cooperate to provide efficient relay. For
example, if the inbound and outbound binding use the same representation for
optimized binary, the implementations MAY cooperate to pass the optimized form
directly from the inbound to the outbound binding. The choice of whether to
implement such cooperation, and if so the means used, is at the discretion of the
binding specification(s) and/or the implementation of the bindings.

Note: a consequence of this architecture is that there are no invariant rules for the degree to which optimizations are preserved as a message passes through intermediaries. Certain outbound bindings may be incapable of any optimization, and will therefore transmit non-optimized forms in all cases. Other bindings may be capable of optimization, but may or may not choose to optimize the same portions (if any) that were optimized in the inbound message. Other bindings, perhaps under the direction of logic provided in SOAP modules or perhaps as consequence of conventions embodied in the bindings, may optimize portions of the message that were not optimized inbound, or which were optimized using different techniques.

3 An Optimized MIME Multipart/Related Serialization of SOAP Messages

3.1 Introduction

The Optimized MIME Multipart/Related Serialization expands upon the Abstract SOAP Transmission Optimization Feature by describing parts of an implementation of this feature using the [XML-binary Optimized Packaging] format as its basis. This specification does not describe a full implementation but is intended to provide support for building a full implementation of the Abstract SOAP Transmission Optimization Feature. In particular, this specification does not specify the use of any transport for the SOAP message. A full implementation based on this specification is described in **4.2 HTTP SOAP Transmission Optimization Feature Name**.

The Optimized MIME Multipart/Related Serialization provides the basis for an implementation of the Abstract Transmission Optimization Feature by describing how to serialize a SOAP envelope in an optimized way, using the [XML-binary Optimized Packaging] format and a MIME Multipart/Related packaging ([RFC 2387]).

More specifically, the SOAP envelope Infoset is transmitted as a MIME Multipart/Related XOP Package (see [XML-binary Optimized Packaging], 4.1 MIME Multipart/Related XOP Packages). Any W3C recommendation-level version of XML is allowed for storing the XOP Infoset created from the SOAP envelope Infoset into the MIME Multipart/Related XOP Package, however, note that the SOAP envelope Infoset MUST be serializable as XML 1.0.

3.2 Serialization of a SOAP message

When sending a SOAP message using the MIME Multipart/Related Serialization, the SOAP envelope Infoset is serialized as specified in [XML-binary Optimized Packaging] 3.1 Creating XOP packages. Specifically:

- The content-type of the outer package MUST be multipart/related.
- The type parameter of the content-type header of the outer package MUST have a value of "application/xop+xml" (see [XML-binary Optimized Packaging], 4.1 MIME Multipart/Related XOP Packages).
- The startinfo parameter of the content-type header of the outer package MUST specify a content-type for the root part of "application/soap+xml".
- The content-type of the root part MUST be application/xop+xml (see [XML-binary Optimized Packaging], 4.1 MIME Multipart/Related XOP Packages).
- The type parameter of the content-type header of the root part MUST specify a content-type of "application/soap+xml".

The result is a MIME Multipart/Related XOP package (see [XML-binary Optimized Packaging]): one body part, the root, containing an XML representation of the modified SOAP envelope, with an additional part used to contain the binary representation of each element that was optimized.

3.3 Deserialization of a SOAP message

When receiving a SOAP message using this Optimized MIME Multipart/Related Serialization, the SOAP Envelope Infoset is reconstructed from the MIME Multipart/Related XOP Package by performing the processing specified in [XML-binary Optimized Packaging] 3.2 Interpreting XOP packages.

Note: conventions or error reporting mechanisms to be used in processing packages that incorrectly purport to be XOP Packages containing a SOAP Envelope are beyond the scope of this specification.

4 HTTP SOAP Transmission Optimization Feature

4.1 Introduction

The HTTP SOAP Transmission Optimization Feature is a binding-level feature implementing the Abstract SOAP Transmission Optimization Feature in an HTTP binding. The basis of this HTTP SOAP Transmission Optimization Feature is the Optimized MIME Multipart/Related Serialization described in **3 An Optimized MIME Multipart/Related Serialization of SOAP Messages**.

This HTTP SOAP Transmission Optimization Feature builds upon the current HTTP binding (see [SOAP Version 1.2 Part 2: Adjuncts] 7. SOAP HTTP Binding), enhancing it with the support of the Abstract SOAP Transmission Optimization Feature. In all aspects not described in this section, the rules of the HTTP binding are not modified.

4.2 HTTP SOAP Transmission Optimization Feature Name

This HTTP SOAP Transmission Optimization Feature is identified by the URI:

- "http://www.w3.org/2004/08/soap/features/http-optimization".

4.3 Implementation

The HTTP SOAP Transmission Optimization Feature uses the Optimized MIME Multipart/Related Serialization (see **3 An Optimized MIME Multipart/Related Serialization of SOAP Messages**) for implementing the Abstract SOAP Transmission Optimization Feature. On the sending side, this feature serializes the SOAP message as described in **3.2 Serialization of a SOAP message** and puts the MIME headers of the resulting MIME Multipart/Related XOP Package in as HTTP headers and the rest of the package into the HTTP body. On the receiving side, this feature extracts the MIME headers from the HTTP headers and the rest of the MIME Multipart/Related XOP Package from the HTTP body and deserializes as described in **3.3 Deserialization of a SOAP message**.

4.3.1 Sending a SOAP message

When sending a SOAP message, the HTTP SOAP Transmission Optimization Feature changes the behavior of [SOAP Version 1.2 Part 2: Adjuncts] 7. SOAP HTTP Binding. This section describes the perturbations to [SOAP Version 1.2 Part 2: Adjuncts] 7.5.1 Behavior of Requesting SOAP Node that result from use of the HTTP SOAP Transmission Optimization Feature. Only those aspects described below differ from the existing operation of the HTTP binding, all other aspects of its operation remaining unchanged.

4.3.1.1 Init

In the "Init" state, a HTTP request is formulated and transmission of the request is initiated. When using the HTTP SOAP Transmission Optimization Feature, the formulation of the request differs from [SOAP Version 1.2 Part 2: Adjuncts] 7.5.1.1 Init as shown in **[HTTP Request Fields]**.

HTTP Request Fields

Field	Value
Content-Type header field	multipart/related
HTTP entity body	SOAP message serialized as described in **3 An Optimized MIME Multipart/Related Serialization of SOAP Messages**

The XOP package is constructed as described in **3 An Optimized MIME Multipart/Related Serialization of SOAP Messages** with the following restriction:

- The XOP Infoset MUST be serialized as `application/xop+xml` in the root part of the package per Section 5 of [XML-binary Optimized Packaging].

- Each optimized *Node* MUST generate exactly one extracted binary part in the resulting package, i.e., extracted binary parts MUST NOT be referenced by more than one `xop:Include` in the SOAP message part.

- Each MIME part that is refered to by `xop:Include` MUST have a Content-Transfer-Encoding header field.

Note: this does not preclude the MIME Multipart/Related package from including additional parts not referenced by a `xop:Include` element. Such additional parts are not part of the SOAP message Infoset and are not included in the SOAP processing model.

Implementations of this binding MUST enforce the restriction that XOP is not to be used with Infosets that contain *element information items* of name `xop:Include` (see [XML-binary Optimized Packaging] 3. XOP Infosets Constructs). In any case where a SOAP envelope containing such an *element information item* is to be sent, the binding MUST do one of the following:

- Fall back to use the `application/soap+xml` media type or any other suitable media type, i.e., send the SOAP envelope without using the HTTP SOAP Transmission Optimization Feature.
- Generate a binding-dependent SOAP fault.

Note that such SOAP envelopes could in principle arise either from data created locally at the sending node or in data relayed at an intermediary. Bindings are responsible for checking all such input as necessary to ensure that the rule just stated is enforced.

4.3.2 Receiving a SOAP message

When receiving a SOAP message, an implementation of the SOAP HTTP Binding (see [SOAP Version 1.2 Part 2: Adjuncts]) will determine whether the HTTP SOAP Transmission Optimization Feature is used by checking the presence of the `application/xop+xml` media type (see [XML-binary Optimized Packaging] Section 5.1). If the media type of the HTTP message is "multipart/related" and the media type of the root part of the MIME Multipart/Related package is `application/xop+xml`, and the start-info parameter indicates a content type of "application/soap+xml" then the received SOAP message was transmitted using the HTTP SOAP Transmission Optimization Feature and MUST be processed accordingly.

The HTTP SOAP Transmission Optimization Feature changes the behavior of [SOAP Version 1.2 Part 2: Adjuncts] 7. SOAP HTTP Binding for the reception of a

SOAP message. The perturbations to [SOAP Version 1.2 Part 2: Adjuncts] 7.5.2 Behavior of Responding SOAP Node that result from use of the HTTP SOAP Transmission Optimization Feature are as follows:

- When making an abstraction of the request message available in `http://www.w3.org/2003/05/soap/mep/InboundMessage`, the HTTP Binding MUST reconstruct the SOAP Envelope Infoset as described in **3.3 Deserialization of a SOAP message**.

All other aspects of the operation of the HTTP binding remain unchanged.

A References

SOAP Version 1.2 Part 1: Messaging Framework
> *SOAP Version 1.2 Part 1: Messaging Framework*, Marc Hadley, Noah Mendelsohn, Jean-Jacques Moreau, *et. al.*, Editors. World Wide Web Consortium, 24 June 2003. This version is http://www.w3.org/TR/2003/REC-soap12-part1-20030624/. The latest version is available at http://www.w3.org/TR/soap12-part1/.

SOAP Version 1.2 Part 2: Adjuncts
> *SOAP Version 1.2 Part 2: Adjuncts*, Henrik Frystyk Nielsen, Noah Mendelsohn, Jean-Jacques Moreau, *et. al.*, Editors. World Wide Web Consortium, 24 June 2003. This version is http://www.w3.org/TR/2003/REC-soap12-part2-20030624/. The latest version is available at http://www.w3.org/TR/soap12-part2/.

XML-binary Optimized Packaging
> *XML-binary Optimized Packaging*, Mark Nottingham, Noah Mendelsohn, Martin Gudgin, and Hervé Ruellan, Editors. World Wide Web Consortium, 25 January 2005. This version is http://www.w3.org/TR/2005/REC-xop10-20050125/. The latest version is available at http://www.w3.org/TR/xop10/.

SOAP 1.2 Attachment Feature
> *SOAP 1.2 Attachment Feature*, Hervé Ruellan and Henrik Frystyk Nielsen, Editors. World Wide Web Consortium, 08 June 2004. This version is http://www.w3.org/TR/2004/NOTE-soap12-af-20040608/. The latest version is available at http://www.w3.org/TR/soap12-af/.

SOAP Optimized Serialization Use Cases and Requirements
> *SOAP Optimized Serialization Use Cases and Requirements*, Tony Graham, Mark Jones, and Anish Karmarkar, Editors. World Wide Web Consortium, 08 June 2004. This version is http://www.w3.org/TR/2004/WD-soap12-os-ucr-20040608/. The latest version is available at http://www.w3.org/TR/soap12-os-ucr/.

Resource Representation SOAP Header Block
> *Resource Representation SOAP Header Block*, Martin Gudgin, Yves Lafon, and Anish Karmarkar, Editors. World Wide Web Consortium, 25 January 2005. This version is http://www.w3.org/TR/2005/REC-soap12-rep-20050125/. The latest version is available at http://www.w3.org/TR/soap12-

rep/.

Assigning Media Types to Binary Data in XML

Assigning Media Types to Binary Data in XML, Ümit Yalçınalp and Anish Karmarkar, Editors. World Wide Web Consortium, 02 November 2004. This version is http://www.w3.org/TR/2004/WD-xml-media-types-20041102. The latest version is available at http://www.w3.org/TR/xml-media-types.

Extensible Markup Language (XML) 1.0 (Third Edition)

Extensible Markup Language (XML) 1.0 (Third Edition), Jean Paoli, Eve Maler, Tim Bray, *et. al.*, Editors. World Wide Web Consortium, 04 February 2004. This version is http://www.w3.org/TR/2004/REC-xml-20040204. The latest version is available at http://www.w3.org/TR/REC-xml.

Namespaces in XML

Namespaces in XML, Andrew Layman, Dave Hollander, and Tim Bray, Editors. World Wide Web Consortium, 14 January 1999. This version is http://www.w3.org/TR/1999/REC-xml-names-19990114. The latest version is available at http://www.w3.org/TR/REC-xml-names.

XML Information Set (Second Edition)

XML Information Set (Second Edition), Richard Tobin and John Cowan, Editors. World Wide Web Consortium, 04 Feb 2004. This version is http://www.w3.org/TR/2004/REC-xml-infoset-20040204. The latest version is available at http://www.w3.org/TR/xml-infoset.

XML Schema Part 1: Structures Second Edition

XML Schema Part 1: Structures Second Edition, David Beech, Murray Maloney, Henry S. Thompson, and Noah Mendelsohn, Editors. World Wide Web Consortium, 28 October 2004. This version is http://www.w3.org/TR/2004/REC-xmlschema-1-20041028/ The latest version is available at http://www.w3.org/TR/xmlschema-1/.

XML Schema Part 2: Datatypes Second Edition

XML Schema Part 2: Datatypes Second Edition, Ashok Malhotra and Paul V. Biron, Editors. World Wide Web Consortium, 28 October 2004. This version is http://www.w3.org/TR/2004/REC-xmlschema-2-20041028/. The latest version is available at http://www.w3.org/TR/xmlschema-2/.

RFC 2119

Key words for use in RFCs to Indicate Requirement Levels, S. Bradner, Editor. IETF, March 1997. This RFC is available at http://www.ietf.org/rfc/rfc2119.txt.

RFC 2387

The MIME Multipart/Related Content-type, E. Levinson, Editor. IETF, August 1998. This RFC is available at http://www.ietf.org/rfc/rfc2387.txt.

RFC 2396

Uniform Resource Identifiers (URI): Generic Syntax, T. Berners-Lee, R. Fielding and L. Masinter, Editors. IETF, August 1998. This RFC is available at http://www.ietf.org/rfc/rfc2396.txt.

RFC 3023

XML Media Types, M. Murata, S. St.Laurent and D. Kohn, Editors. IETF, January 2001. This RFC is available at http://www.ietf.org/rfc/rfc3023.txt.

PASWA

Proposed Infoset Addendum to SOAP Messages with Attachments, AT&T, BEA Systems, Canon, Microsoft Corporation, SAP AG and Tibco Software, April 2003. This document is available at http://www.gotdotnet.com/team/jeffsch/paswa/paswa61.html.

B Acknowledgements (Non-Normative)

This specification is the work of the W3C XML Protocol Working Group.

Participants in the Working Group are (at the time of writing, and by alphabetical order): David Fallside (IBM), Tony Graham (Sun Microsystems), Martin Gudgin (Microsoft Corporation, formerly of DevelopMentor), Marc Hadley (Sun Microsystems), Gerd Hoelzing (SAP AG), John Ibbotson (IBM), Anish Karmarkar (Oracle), Suresh Kodichath (IONA Technologies), Yves Lafon (W3C), Michael Mahan (Nokia), Noah Mendelsohn (IBM, formerly of Lotus Development), Jeff Mischkinsky (Oracle), Jean-Jacques Moreau (Canon), Mark Nottingham (BEA Systems, formerly of Akamai Technologies), David Orchard (BEA Systems, formerly of Jamcracker), Herve Ruellan (Canon), Jeff Schlimmer (Microsoft Corporation), Pete Wenzel (SeeBeyond), Volker Wiechers (SAP AG).

Previous participants were: Yasser alSafadi (Philips Research), Bill Anderson (Xerox), Vidur Apparao (Netscape), Camilo Arbelaez (webMethods), Mark Baker (Idokorro Mobile, Inc., formerly of Sun Microsystems), Philippe Bedu (EDF (Electricite De France)), Olivier Boudeville (EDF (Electricite De France)), Carine Bournez (W3C), Don Box (Microsoft Corporation, formerly of DevelopMentor), Tom Breuel (Xerox), Dick Brooks (Group 8760), Winston Bumpus (Novell, Inc.), David Burdett (Commerce One), Charles Campbell (Informix Software), Alex Ceponkus (Bowstreet), Michael Champion (Software AG), David Chappell (Sonic Software), Miles Chaston (Epicentric), David Clay (Oracle), David Cleary (Progress Software), Dave Cleary (webMethods), Ugo Corda (Xerox), Paul Cotton (Microsoft Corporation), Fransisco Cubera (IBM), Jim d'Augustine (Excelon Corporation), Ron Daniel (Interwoven), Glen Daniels (Macromedia), Doug Davis (IBM), Ray Denenberg (Library of Congress), Paul Denning (MITRE Corporation), Frank DeRose (TIBCO Software, Inc.), Mike Dierken (DataChannel), Andrew Eisenberg (Progress Software), Brian Eisenberg (DataChannel), Colleen Evans (Sonic Software), John Evdemon (XMLSolutions), David Ezell (Hewlett Packard), James Falek (TIBCO Software, Inc.), Eric Fedok (Active Data Exchange), Chris Ferris (Sun Microsystems), Daniela Florescu (Propel), Dan Frantz (BEA Systems), Michael Freeman (Engenia Software), Dietmar Gaertner (Software AG), Scott Golubock (Epicentric), Mike Greenberg (IONA Technologies), Rich Greenfield (Library of Congress), Hugo Haas (W3C), Mark Hale (Interwoven), Randy Hall (Intel), Bjoern Heckel (Epicentric), Frederick Hirsch (Zolera Systems), Erin Hoffmann (Tradia Inc.), Steve Hole (MessagingDirect Ltd.), Mary Holstege (Calico Commerce), Jim Hughes (Fujitsu Limited), Oisin Hurley (IONA Technologies), Yin-Leng Husband (Hewlett Packard, formerly of Compaq), Ryuji Inoue (Matsushita Electric Industrial Co., Ltd.), Scott Isaacson (Novell, Inc.), Kazunori Iwasa (Fujitsu